Model Development Report

dataMineR

30 augustus 2013

CONTENTS

CHAPTER 1

INTRODUCTION

This analysis report is generated using R, R-studio and knitr to knitr R code from markdown into html and later LateX format. We have the option to include all R code that is used to generate the plots and calculations. By default this feauture is dissabled.

The model development step is the third step in a datamining analysis. Steps identified in the datamining process are:

- Data analysis
- Behaviour analysis
- Missing value analysis
- Missing value imputation (optional)
- Binning
- Feature selection
- Model development
- Model analysis
- Model deployment

For the model development process there are a lot of ttechniques and algorithms available in R. For now we will focus on randomForest implemented in the package randomForest() in R.

Other options are logistic regression or adaBoost which is arguably the “best of the shelf classifier in the world”.

1.1 Information on Dataset

Basic information from the dataset we are using.

We are using data from file : ../data/model-set.Rdata. The dataset has 26 variables and 92467 rows.

CHAPTER 2

VALIDATION SETUP

The basic challengence in datamining is that we will develop a predictive model that has predictive capability on onseen data. There are a lot of learning methods that can leverage a large proportion of the collected data using n-fold cross validation.

If there are enough cases we can use a simple shema that holds back a percentage of cases as a test set. The training set will be used entirely for model development using the above mentioned validation methods. The test set is only used in model evaluation.

We have a train set of 64726 cases. The test set contains 27741 cases.

The test set is saved in ../data/test-set.Rdata.

CHAPTER 3

FEATURE SELECTION

As we often have a big dataset with a lot of cases and a lot of variables(or features) we will have to implement a feature selection method to reduce the number of variables to be used in the final model. This is to prefent issues with colinearity and overfitting.

To do this we have to get an idea on the importance of variables in relation to the prediction task. The random forest package has a function to assess variable importance in a model set. This function build repeatedly a random forest leaving one variable out of the model set. Then each model is tested on the Out Of Bag set and the decrease in gini is captured. The variable with the highest decrease in gini is the most valuable.

	MeanDecreaseGini
action	46.46
product_detail	43.91
contact_channel	40.73
months_under_contract	35.52
sales_channel	33.08
product_grp	15.03
days_since_last_sale	12.89
index1	11.26
days_since_last_contract_adj	10.77
usage	8.835
unkown_amount	7.909
usage2	7.23
age	7.048

donwpayment	7.042
days_since_last_sale_p1	6.752
P1_turnover	6.718
extra_payment	6.715
days_since_last_sale_p2	6.598
P2_turnover	6.511
tax_amount	5.584
city_code	5.543
P1_amount	5.211
contract_type	4.697
P2_amount	4.634
customer_lifetime	4.243

3.1 Unimportant Variables

Based on the initial assessment of variable importance we can see that variables for which no significant (< 1.0) decrease in the Gini index is detected can savely be removed from the modelling set.

| MeanDecreaseGini | |:——————:| | |

CHAPTER 4

TREES

We start by building a simple Decision Tree model using the rpart package. For classification a recursive partitioning sheme is used. The split criterium uses the Gini index to calculate the best split on each node.

Let us take a look at a simple tree using the two most important variables.

In this plot the width of the branches denotes the percentage of cases falling in that branch. In the leafs we see the change that churn occurs in that leaf. This result is emphasized by the color the leaf has, red means high change of churn , green low change of churn.

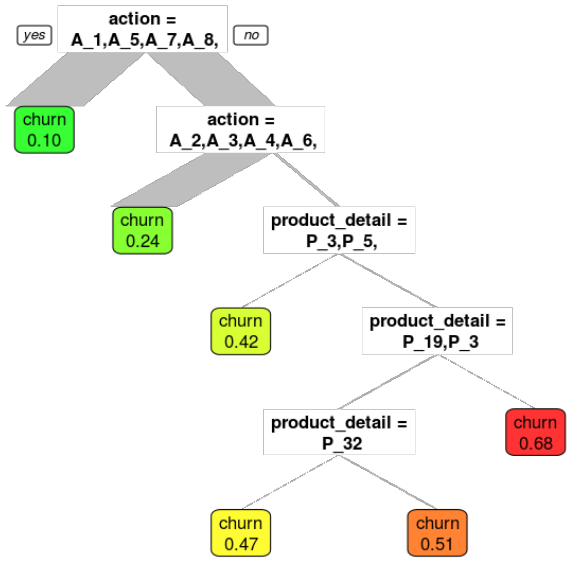
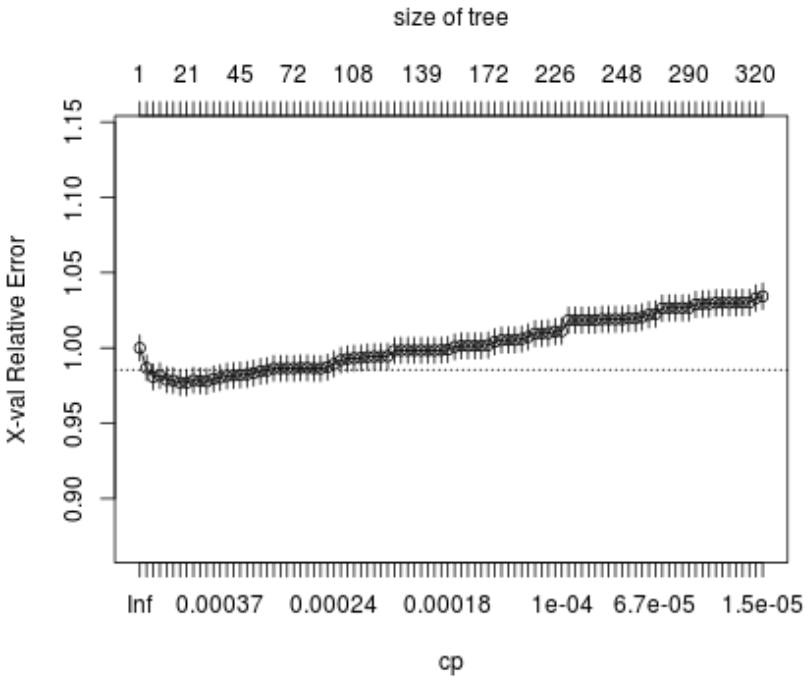


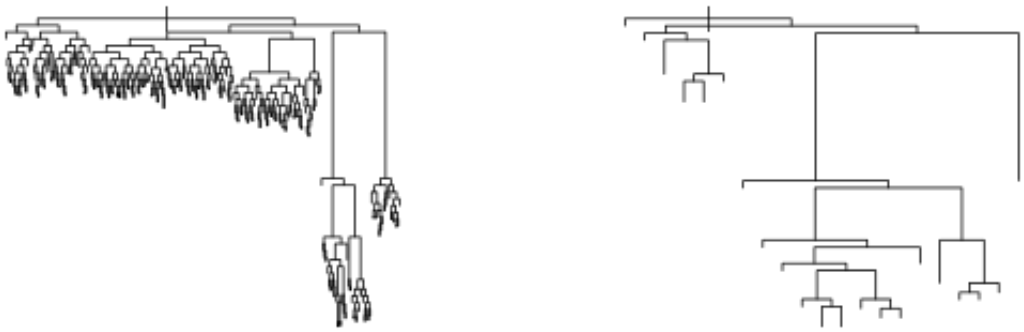
Figure 4.1: Simple decision tree

4.1 Pruning the tree

Usually a full tree is build, which is pruned back to a size that is optimal given the information available in the dataset. This can be done by checking the complexity factor cp. More on pruning methodologies eand techniques here.



In the above figure we can see how the relative error on the cross validated trees increases as the tree size gets bigger. This indicated overfitting.



A good tree size is the size for which the complexity factor is minimal. In this case the minimal cp is 4.5578 × 10-4.

4.2 Optional : Cost matrix

We can specify different cost for false negatives (FN) and false positives(FP)

CHAPTER 5

FORESTS

Random forests are an ensemble learning method for classification (and regression) that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. More on wikipedia here.

Random Forests are implemented in R by the package randomForest.

5.1 Parameters

Number of trees per model

Node size

5.2 Stratified dataset

This modelling technique works best on stratified datasets.

5.3 Use parallel computing for big datasets

As we have more then 100k cases we use the package foreach and multicore to build forests in parallel and combine results afterwards.

```
## [1] "doMC"
```

```
## [1] 4
```

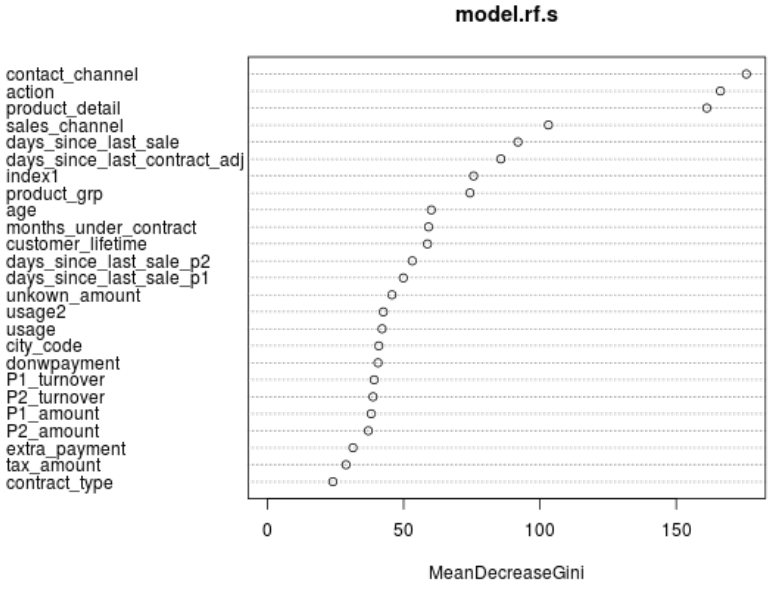


Figure 5.1: Variable Importance Stratified random forest

CHAPTER 6

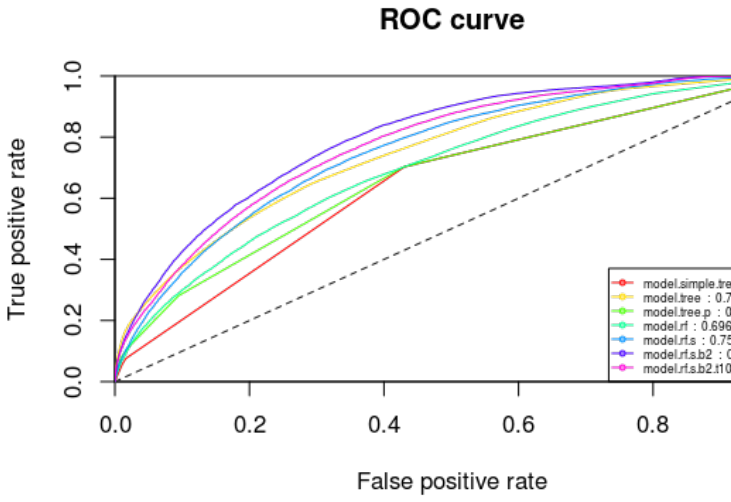
MODEL QUALITY

We have build some trees and forests.

The following models will be evaluated in this section: model.simple.tree, model.tree, model.tree.p, model.rf, model.rf.s, model.rf.s.b2, model.rf.s.b2.t10

Results can be captured in a couple of ways.

6.1 Area under the Reciever Operating Curve



Plot ROC curves and AUC statistics on training set.

6.2 Koglomoroff Smirnov Statistic

Seperation between cumulative good and bad.

6.3 Rank correlation

todo

6.4 Score table

Prediction on the training set result shown in equal volume bands. Average churn rate in a band is used as the predicted probability of the target behaviour.

No	Yes	n	p	p.lcl	p.ucl
6447	118	6565	0.01797	0.0149	0.02149
6158	283	6441	0.04394	0.03906	0.04923
5984	429	6413	0.0669	0.0609	0.07329
5923	564	6487	0.08694	0.0802	0.09406
5739	804	6543	0.1229	0.115	0.1311
5492	1014	6506	0.1559	0.1471	0.1649
5208	1263	6471	0.1952	0.1856	0.205
4768	1589	6357	0.25	0.2394	0.2608
4430	2057	6487	0.3171	0.3058	0.3286
3477	2979	6456	0.4614	0.4492	0.4737

The model selected is model.rf.s, described as Random Forest on stratiefied dataset.

CHAPTER 7

SAVE MODELS

Save models that where developped in this step.

Models saved for evaluation are listen in the table below:

model.simple.tree, *model.tree*, *model.tree.p*, *model.rf*, *model.rf.s*, *model.rf.s.b2* and *model.rf.s.b2.t10*

CHAPTER 8

NEXT STEP

The next step in model evaluation on the test set.